SLAM Algorithms

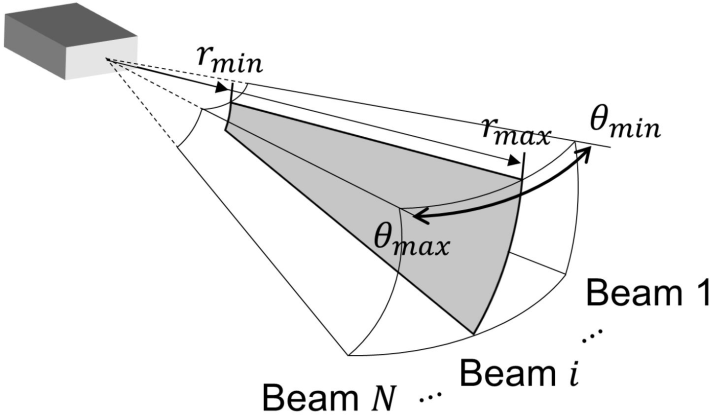
EDAA - G06

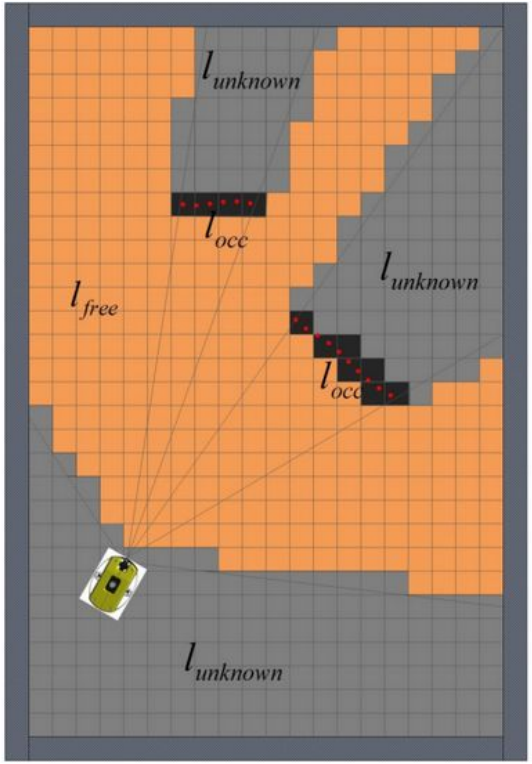
Tiago Duarte
João Costa – João Martins
Henrique Ribeiro

3D Mapping

Sonar Data & 2D Mapping

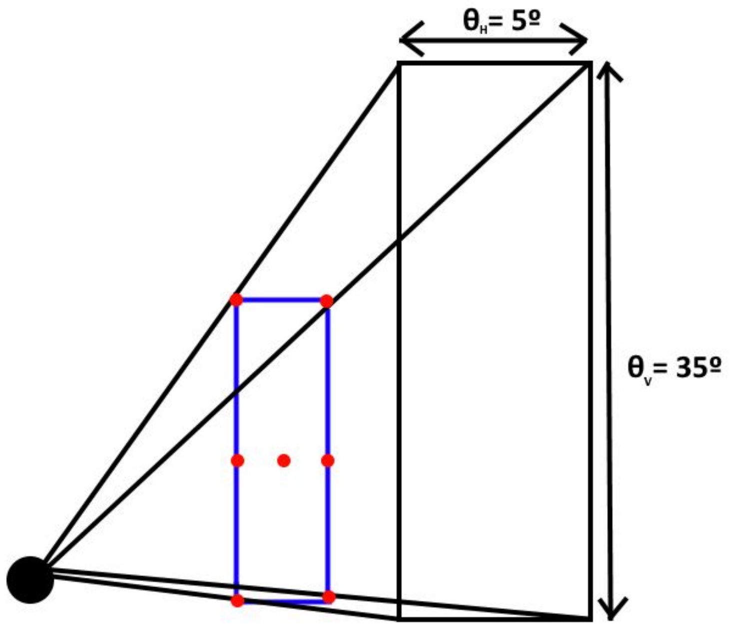
- Sonar rotates around itself
- Sends/measures waves in a cone
- Each beam has multiple intensities across several intervals



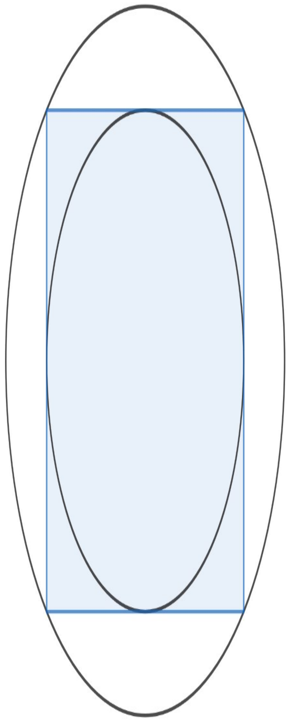


3D - How to Cover a Volume?

- Estimate covered cells through 3D raycasts
- Choose destination points to achieve maximum coverage
- Our algorithm, adapted from Fula[0] will cast to 7 points



Approximating the Sonar as a Rectangle



Problem: If we update the bounding rect of the 5º sonar angle, we would map possibly occupied space as empty

Solution: By using an angle reduced by a constant factor, it's bounding box will be circumscribed within the origin sonar's beam width

OctoMap Coverage

The sonar's range is limited to 5m:

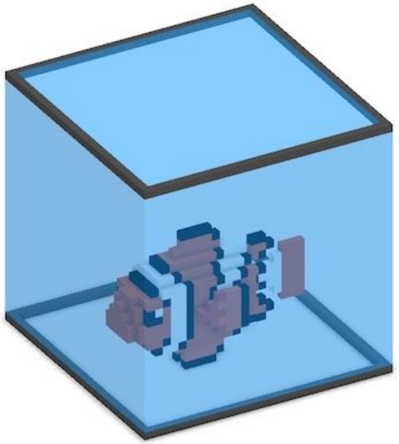
7 raycasts will be sufficient

Let's assume a larger range:

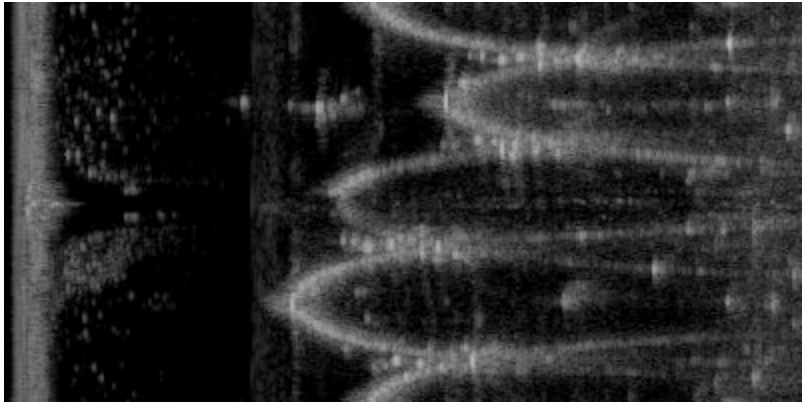
The entire volume **will not be updated**

We can increase the number of raycasts:

Tradeoff between **efficiency** and **precision**



Range to First Feature

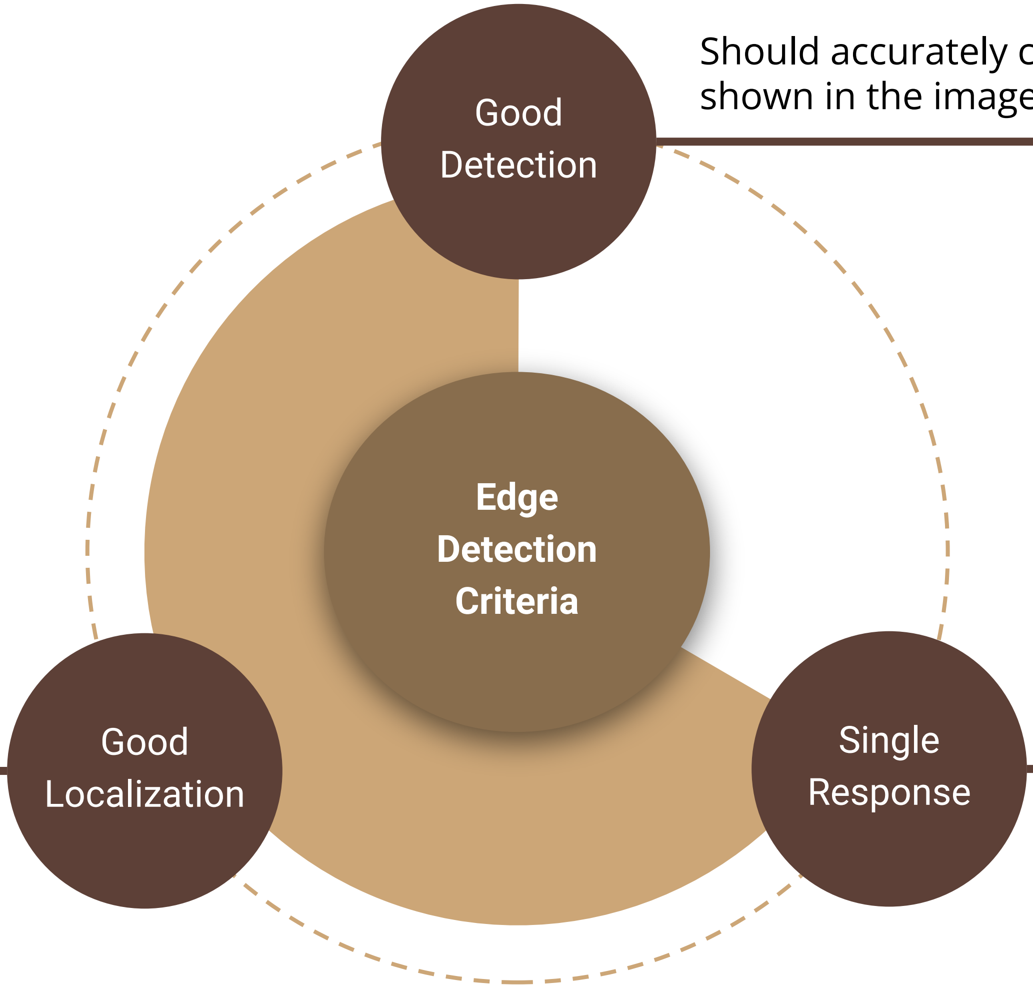


I. Ignore first few measurements
II. Apply an intensity threshold
III. Apply an edge detector
IV. Select first value higher than a dynamic threshold T_d

Edge Detection

Low error rate

Should accurately catch as many edges shown in the image as possible



Each edge should only be marked once

Minimize edges created from noise

Edge points should be localized in the centre of the edge

Canny Edge Detection

Image Smoothing	Differentiation	Non-Maximum Suppression	Thresholding & Hysteresis
Smooth image using a Gaussian function, with width σ	Calculate the magnitude and angle of the gradient	Reduce thickness of edges and localize their centre	Filter relevant edges using two thresholds and reduce streaking

Canny results

[1] Image Smoothing

Convert the image into **grayscale**, then **convolve** the image with a **Gaussian filter** to smooth noise

$$\mathbf{B} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

Usually, a 5x5 kernel is used, but its standard deviation can usually be altered

In the kernel above, σ = 1

[2] Differentiation

Calculate the first derivative in the horizontal G_x and vertical G_Y directions using an **edge detector operator** such as:

- Robert's Cross
- Prewitt's Operator
- Sobel's Operator

$$\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

$$\mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

[2] Differentiation

To characterize the gradient, two matrixes are created to store its **Intensity** and **Angle**.

$$\mathbf{G} = \sqrt{{\mathbf{G}_x}^2 + {\mathbf{G}_y}^2}$$

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

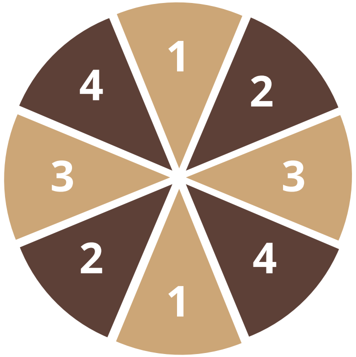
[3] Non-Maximum Suppression

To ensure good localization, we must apply **edge thinning** to remove unwanted points, ideally resulting in one-pixel edges

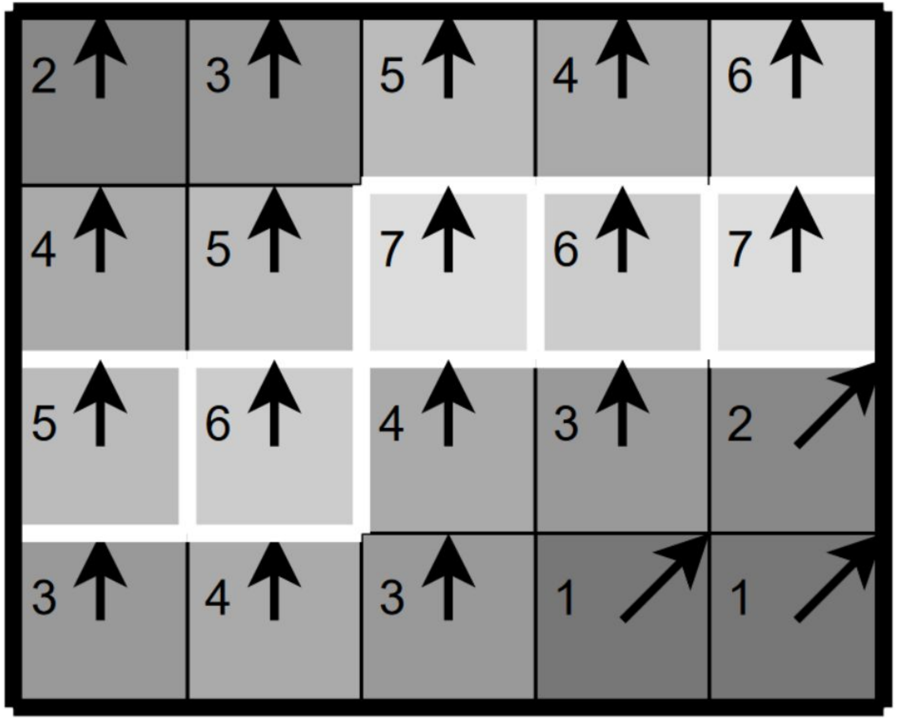
To achieve this, Non-Maximum Suppression removes all points that are not the local maxima of its respective edge

A simple implementation discretizes the gradient's angle into an 8-connectec neighbourhood, and then compare its intensity with the two cells in its direction

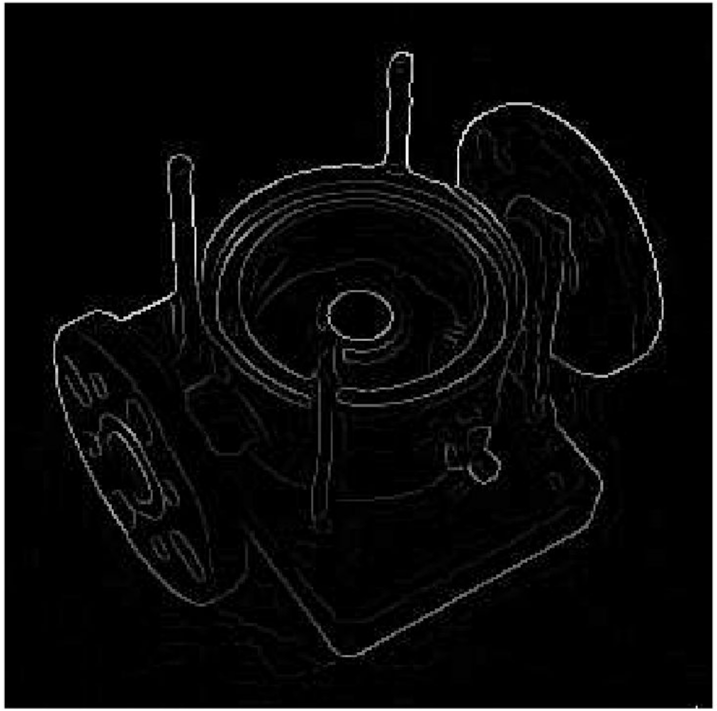
If the cell's value is higher, its intensity is kept, otherwise the value is discarded



[3] Non-Maximum Suppression



[3] Non-Maximum Suppression



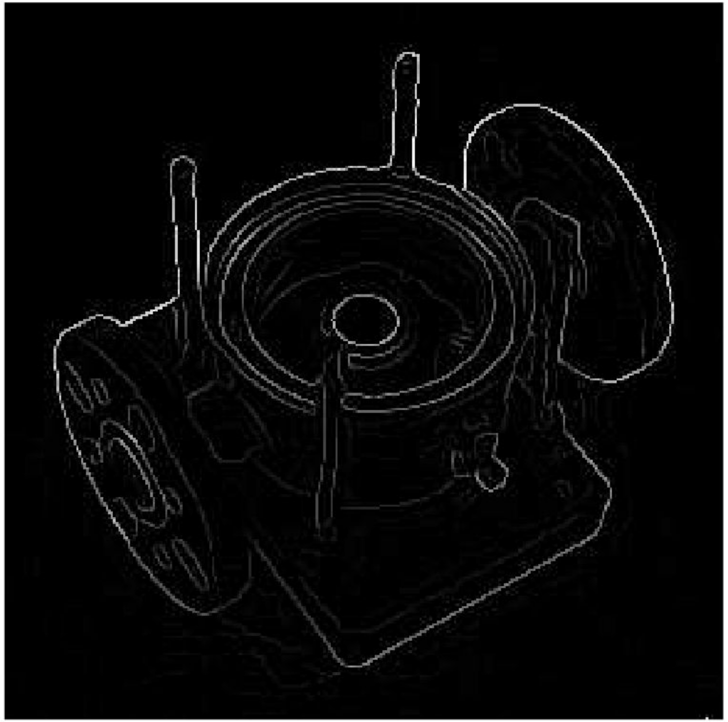
[4] Threshold & Hysteresis

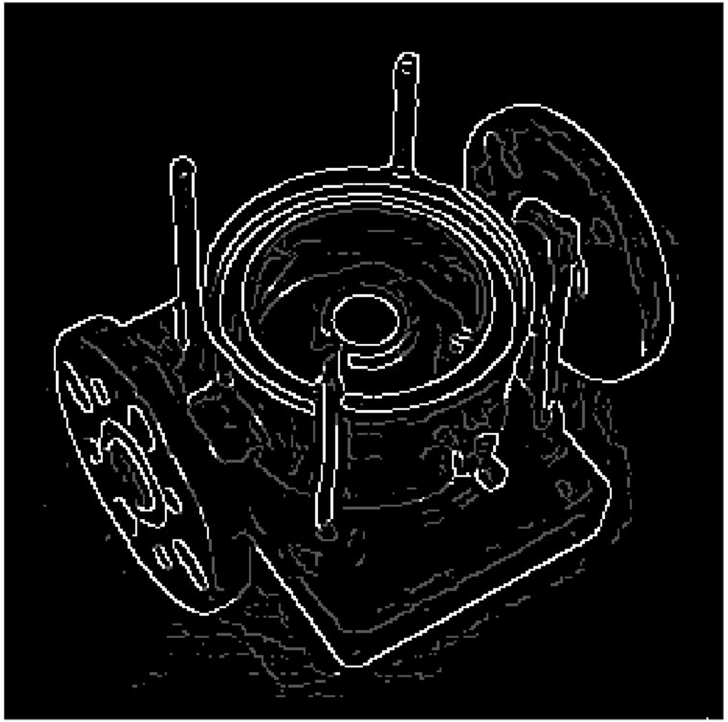
A single threshold limit usually causes edge values to fluctuate above and below this value, making the line appear broken, called **'streaking**'.

By using two thresholds, we can classify a pixel in 3 categories:

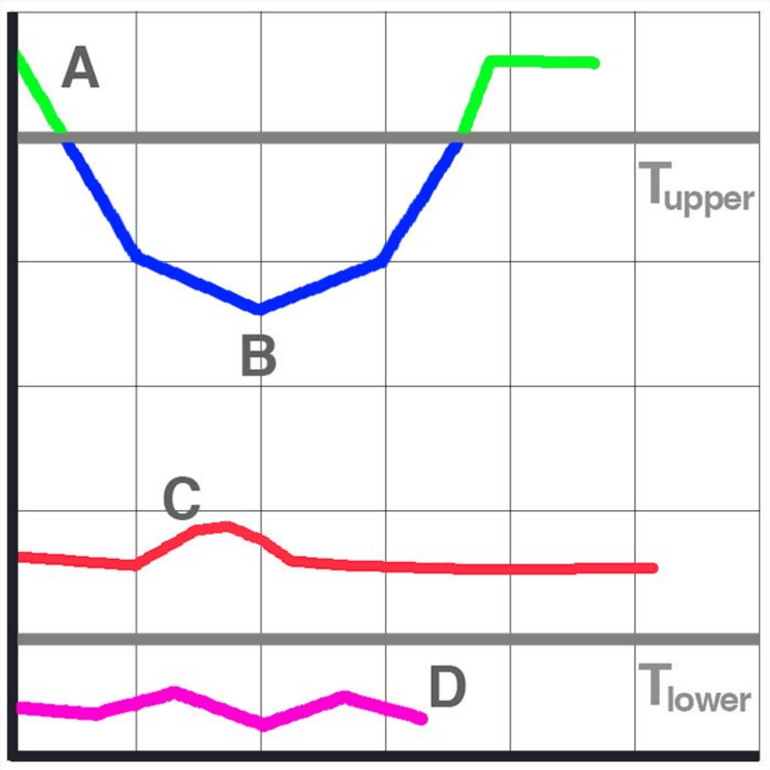
Upper Threshold	**Strong**	Definitely an edge
	Weak	Edge candidates
Lower Threshold	**Ignored**	Never an edge

Strong vs Weak edges





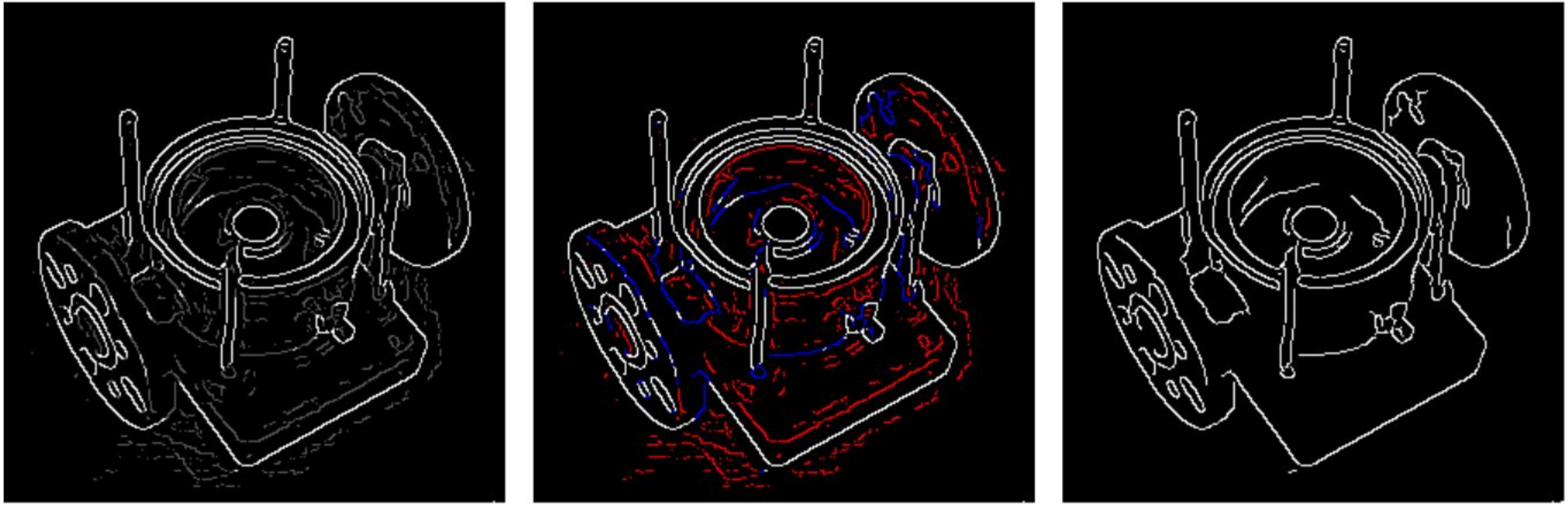
Threshold & Hysteresis



A	Always an Edge
B	Edge
C	Not an edge
D	Never an edge

Weak edges are only kept if they are connected to a strong edge

Hysteresis Result



How to determine if a weak pixel belongs to an edge with at least one strong pixel?

Connected Component Labelling

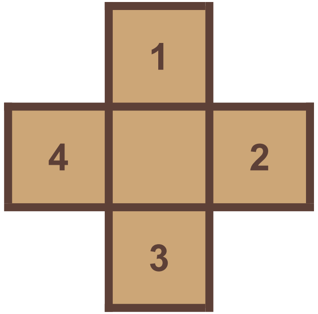
or Blob Analysis

Connected Component Labelling

Goal

- Label each connected region in a binary image, often called blobs, with the same unique label

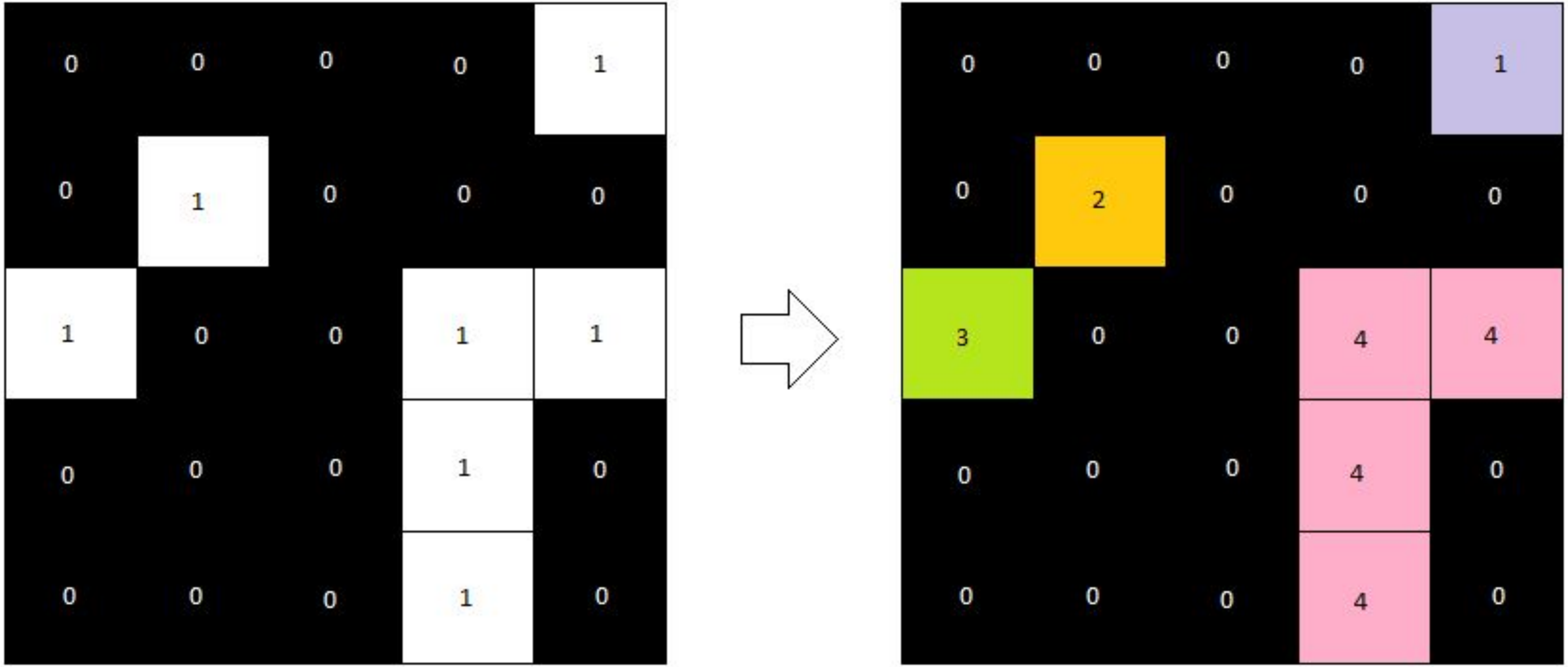
Different Connectivities



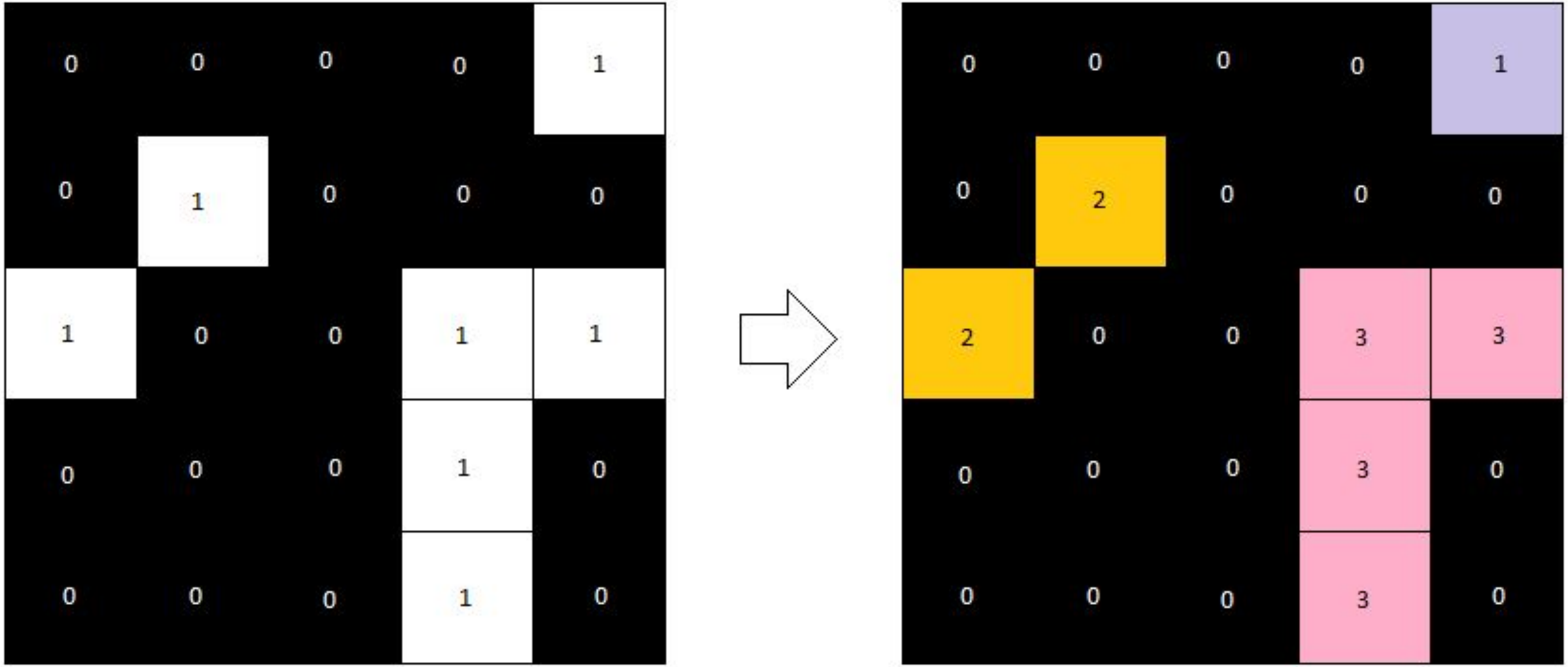
4-way connectivity

8-way connectivity

4-Way Connectivity

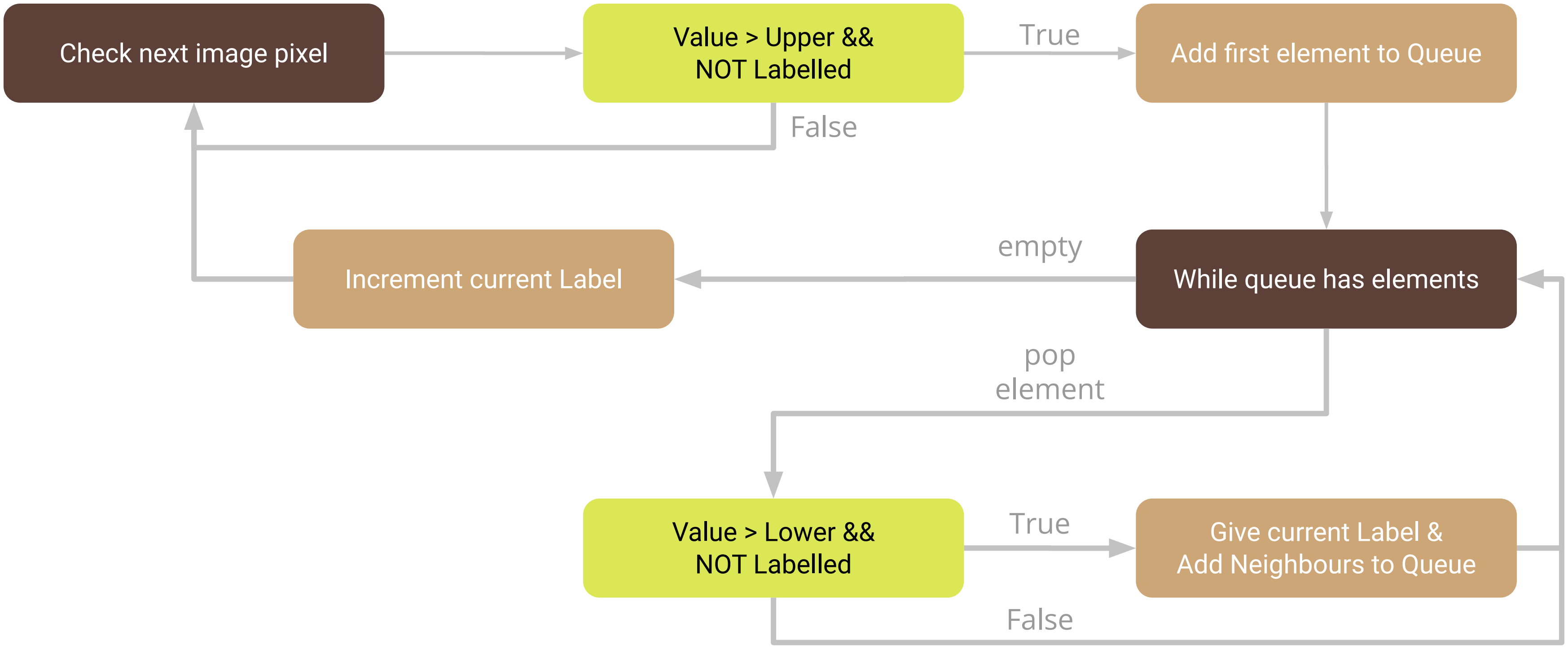


8-Way Connectivity

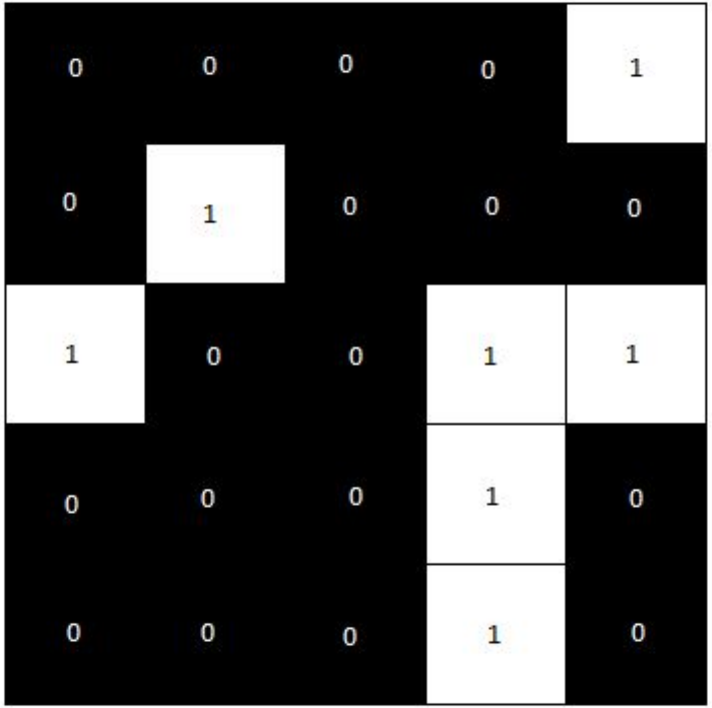


One Component at a Time Algorithm

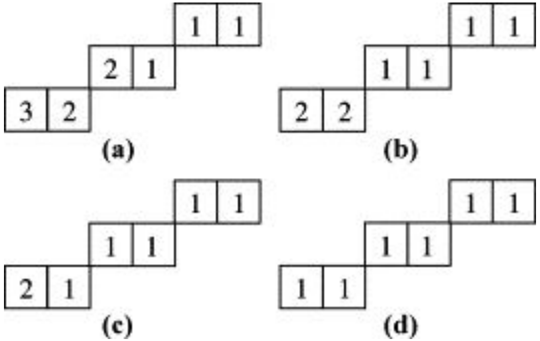
One Component At a Time

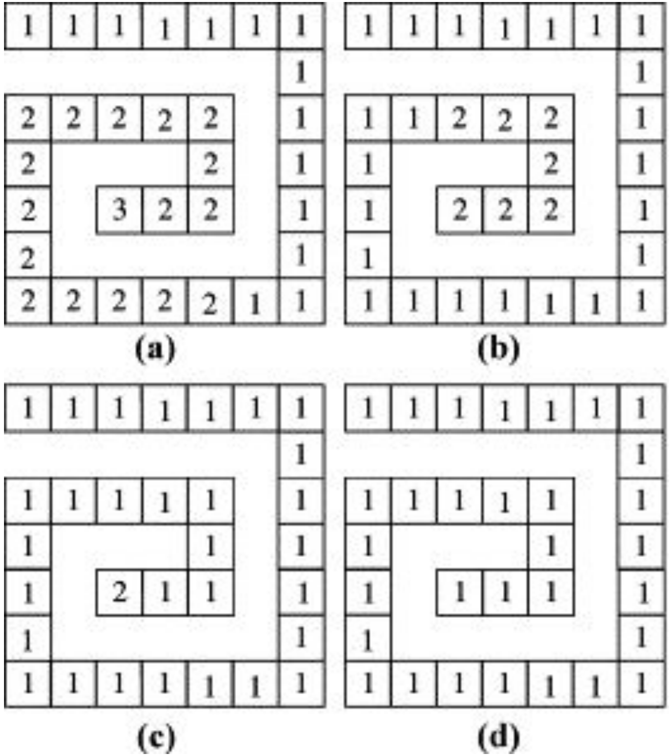


One Component at a Time



Other variations





Two Pass Algorithm

First Pass

For each non-zero pixel, check its neighbours:

No non-zero neighbours		Give a new Label as it is a new component
One non-zero neighbour		Give the same label
Many non-zero neighbours	**Same labels**	Give the same label
	Different Labels	Set current pixel to the lowest label Add a note to the equivalences table

Second Pass

For each pixel with a label, check the equivalences table and update its value

Problem

How to implement this equivalences table?

- Hashmap
- Disjoint Sets

Applying to the Edge Detection Problem

A) Thresholded image is not binary

Naive: Assume weak and strong pixels as non-zero

Problem: Loses weak vs strong information

B) Keep only pixels which belong to a strong edge

Naive: Third pass to mark if each label is a strong edge

Problem: Adding another pass hinders performance

Proposed Solution: Weak & Strong Labels

Change the label numbering according to the pixel type in the source image

Strong

Each new strong label is given incrementally from 0

Weak

Each new weak label is given decrementally from MAXINT

Weak Label: Label is higher than the lowest assigned weak label
Strong Label: Label is lower than the highest assigned strong label

Updated rules of the first pass:

- When processing a strong pixel, instead of simply copying the label from its neighbours, if the label to be assigned belongs to a weak label, create a new strong label instead and add a new entry to the equivalences table

3D Mapping

Range to First Feature

(adapted)

I. Ignore first few measurements

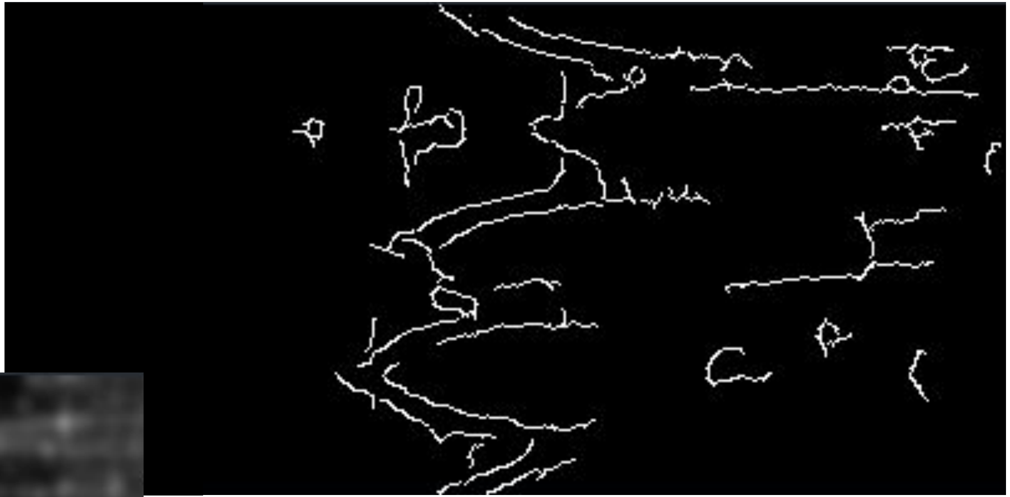
~~II. Apply an intensity threshold~~

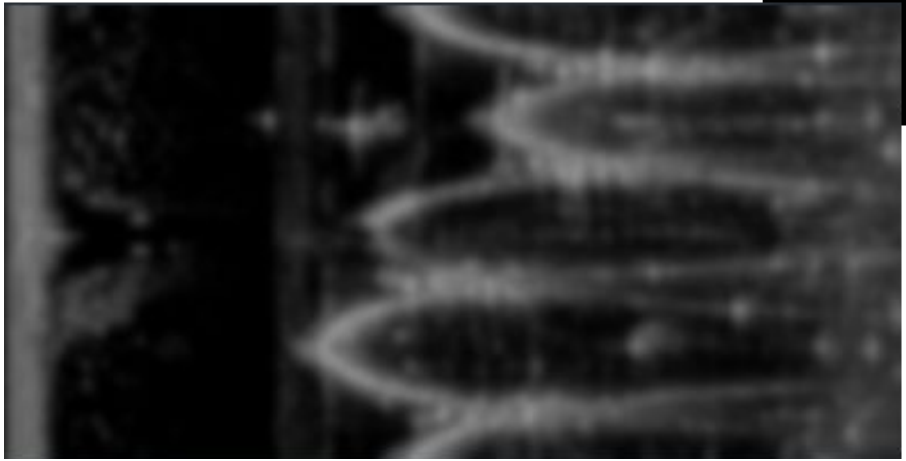
III. Apply Canny's Edge Detector

IV. Select first non-zero value

V. Find local Maxima in original

Results





Questions?

References

[0] - João Pedro Bastos Fula - Underwater mapping using a SONAR
https://en.wikipedia.org/wiki/Canny_edge_detector
https://en.wikipedia.org/wiki/Connected-component_labeling
https://en.wikipedia.org/wiki/Deriche_edge_detector
https://moodle.up.pt/pluginfile.php/183979/mod_resource/content/13/VCOM_04%20-%20Edge%20and%20Line%20Detection.pdf

https://towardsdatascience.com/implementing-a-connected-component-labeling-algorithm-from-scratch-94e1636554f
http://kiwi.bridgeport.edu/cpeg585/CannyEdgeDetector.pdf

https://www.sciencedirect.com/science/article/abs/pii/S1077314202000309?via%3Dihub

Slide deck source

Slide deck